

METHOD AND APPARATUS FOR EXCHANGING SUB-HIERARCHICAL STRUCTURES WITHIN A HIERARCHICAL FILE SYSTEM

FIELD OF THE INVENTION

[0001] The present invention generally relates to managing a plurality of files and directories within a hierarchical file system, and in particular to a system and method for exchanging a first sub-hierarchy having a root directory coinciding with the root directory of a hierarchical file system with a sub-hierarchy included in the hierarchical file system.

BACKGROUND OF THE INVENTION

[0002] A computer system is capable of storing large amounts of information, particularly on large capacity permanent storage media known as hard drives. File systems, particularly hierarchical file systems (HFSs), are currently widely used for providing organization of stored information so that desired information can be easily and quickly located and retrieved. In an HFS, user data is organized into variably sized units known as files. Files are then organized into groups known as directories. Directories may contain files and may also contain other directories, known as subdirectories. This structure forms a branching tree, or hierarchy. The topmost node of this tree is a special directory known as the root directory. The hard drive is divided into physical sections known as sectors. The hard drive can also be divided into one or more logical groups of sectors known as partitions. The HFS provides a mapping between disk sectors and user data, and maintains internal housekeeping information for storing the structural relationship between directories, sector usage tables, etc.

[0003] The file systems are typically stored on non-archival rotating storage media which are subject to mechanical failures of the media and/or drive thereof, endangering the viability of the data stored therein. In addition, inadvertent or malicious failures may occur in software within the file system itself (e.g., corruption) and/or data stored therein.

[0004] The operating system of the computer is typically provided with one or more file system drivers for enabling the operating system to interpret the file system's mapping for accessing and/or processing data stored within the file system. When a file system fails or becomes corrupted, the operating system may be unable, partially or completely, to interpret the file system. Furthermore, the operating system's data and executable code is typically stored in files of the file system, such that viability of the operating system is endangered when the file system fails or is corrupted.

[0005] A popular method for protection of stored data is copying (i.e., backing up) the file system to other media (i.e., backup) and recovering from a corruption occurrence by restoring the corrupted file system from the backup data. In both of these cases, the user is typically inconvenienced either during the backup process or the recovery process, since the computer system may not be useable during that time. Furthermore, the backup media may not be available when a recovery process is necessary. In addition, once the recovery process has been initiated, it is usually not possible to cancel or reverse the recovery.

[0006] In another popular method for protection against corruption within a file system, operating systems typically provide repair tools for checking the file system and performing repairs. If the operating system is itself stored on the corrupted file system, a

situation may develop in which the operating system cannot be started because of the corruption, preventing use of the repair tool to repair the corruption.

[0007] Typically the repair tool returns the file system to a default condition, often overwriting or deleting files which are desirable to be kept, or making these files inaccessible by deleting the applications needed to access them. Furthermore, the standard repair tool typically checks and repairs a file system which resides in one partition. Backup data outside of the file system, such as backup data stored on another partition or medium, are not typically checked for corruption or repaired by the repair tool.

[0008] In order to increase speed and transparency for the backup or recovery processes, it is desirable to store one or more sets of backup data on the same disk that stores the original file system. Recently, such backup systems have been developed. However, a recovery process typically includes replacing the corrupted file system, in which the current version of the operating system resides, with a backed up file system having a backed up version of the operating system, without the ability to replace selected portions of the file system. Accordingly, uncorrupted portions of the operating system, application programs and user data which resided in the corrupted file system are replaced by corresponding backup data obtained at a previous point in time, causing a loss of updated information.

[0009] Another disadvantage to such existing backup systems is the requirement for a special device driver(s) for the backup medium, limiting compatibility between available different storage devices and the operating system.

[0010] Furthermore, one class of solution of existing backup systems copies the backup file system onto the same medium as the original file system, however the original and backup file systems are each stored in different respective partitions, causing the waste of unused disk space in each partition as well as lack of continuous monitoring and maintenance of the backup data by the repair tool.

[0011] It is an object of the present invention to provide a method and apparatus for maintaining multiple copies of a file system's data within the file system, and for replacing selectable portions of data from a copy currently being used as a working file system with corresponding data from a different copy by manipulating the file system's internal housekeeping information.

[0012] It is another object of this invention to provide a method of using an individual copy of the file system's data without affecting corresponding portions in another copy.

[0013] It is another object of the present invention to store the multiple copies of the file system in a single partition of a single physical or virtual storage medium.

[0014] It is still another object of the present invention to exchange an individual copy of the multiple copies of the file system's data that is being used as a working file system with another copy.

[0015] Finally, it is an object of the present invention to exchange an individual copy of the multiple copies of the file system's data with another copy, where the replacement is reversible.

SUMMARY OF THE INVENTION

[0016] A system and method is provided for exchanging a first sub-hierarchy of at least two sub-hierarchies of a hierarchical file system (HFS) with a second sub-hierarchy of the at least two sub-hierarchies, the HFS being accessible by at least one processor and having a root directory that is a parentless directory, the method including the steps of providing for the first sub-hierarchy to include a first root directory stored in a first location used for (i.e., occupied by) the root directory of the HFS and a first plurality of files configured to branch therefrom; providing for the second sub-hierarchy to include a second root directory stored in a second location of the HFS that is not used (i.e., occupied by) by the root directory of the HFS and a second plurality of files configured to branch therefrom; and providing for relocation of the second root directory to the first location. Preferably, the method further includes the steps of providing for configuration of the second plurality of files to branch from the second root directory including while the second root directory is located in the first location. Preferably, the method further includes the steps of providing for relocation of the first root directory to the second location and providing for configuration of the first plurality of files to branch from the first root directory including while the first root directory is located in the second location.

[0017] The present invention further provides a computer system including a processor accessible hierarchical file system (HFS) having at least two sub-hierarchies including first and second sub-hierarchies and a parentless root directory, wherein the first sub-hierarchy includes a first root directory stored in a first location used for (i.e., occupied by) the root directory of the HFS and a plurality of files configured to branch therefrom,

and the second sub-hierarchy includes a second root directory stored in a second location of the HFS different from the first location and a second plurality of files configured to branch therefrom; and at least one device for moving the second root directory into the first location and configuring the second plurality of files to branch therefrom. Preferably, the at least one device further moves the first root directory into the second location and configures the first plurality of files to branch therefrom.

[0018] The invention further includes a computer readable medium storing a set of programmable instructions configured for execution by the at least one processor, the programmable instructions including means for providing for the HFS including the first and second sub-hierarchies, means for providing for configuration of the second plurality of files to branch from the first location, and means for providing for relocation of the second root directory to the first location. Preferably, the programmable instructions further include means for providing for configuration of the first plurality of files to branch from the second location; and means for relocation of the first root directory to the second location.

[0019] The invention further includes a computer data signal embodied in a transmission medium for execution, the signal including a code segment including instructions for providing for the HFS including the first and second sub-hierarchies, a code segment including instructions for configuring the second plurality of files to branch from the first location; and a code segment including instructions for relocating the second root directory to the first location. Preferably, the signal further includes a code segment including instructions for configuring the first plurality of files to branch from

the second location; and a code segment including instructions for relocating the first root directory to the second location.

BRIEF DESCRIPTION OF THE DRAWINGS

[0020] Various embodiments of the invention will be described herein below with reference to the figures wherein:

[0021] FIG. 1 is a block diagram of a computer system in accordance with the present invention;

[0022] FIG. 2A is a tree diagram of an hierarchical file system in a first state in accordance with the present invention;

[0023] FIG. 2B is a tree diagram of the hierarchical file system of FIG. 2A in a second state;

[0024] FIG. 3 is a flowchart showing procedural steps of a method for exchanging sub-hierarchies in accordance with the present invention;

[0025] FIG. 4 is a flowchart of a first embodiment of the method shown in FIG. 3; and

[0026] FIG. 5 is a flowchart of a second embodiment of the method shown in FIG. 3.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0027] In the present invention, a method and system are provided for changing the configuration of a single hierarchical file system (HFS) by exchanging first and second sub-hierarchies of at least two sub-hierarchies included within the HFS so that before the exchange procedure the first sub-hierarchy is a working file system of the HFS, as described further below, and after the exchange procedure the second sub-hierarchy is the

working file system of the HFS, where a root directory of the working file system is located at the root directory of the HFS, and where the root directory of the HFS is the top most node of the HFS, which typically does not have a parent node (directory, file or equivalent), i.e., the HFS is not included in a different hierarchical file system, and/or the root directory of the HFS is the path through which a currently executing operating system accesses the HFS. Preferably the HFS further includes special files, which may include files and/or directories, which are accessible via the root directory and by the current working file system.

[0028] With reference to FIG. 1, a computer system designated generally by reference numeral 10 is shown having at least one processor 12 accessing at least one data storage unit 14. The at least one data storage unit 14 typically includes a rotating disk, also known as a hard drive storing data in a HFS formation implemented in HFS implementations, such as FAT32 or NTFS. The HFS formation includes a root area through which other areas of the HFS are accessible by direct or indirect connection and/or containment. The at least one data storage unit 14 may span several storage devices, yet form a single volume. The computer system 10 further includes a file system exchange software (FSES) module 16 including a series of programmable instructions executable by the at least one processor 12 for implementing the exchange of sub-hierarchies, as described in greater detail below.

[0029] The at least two sub-hierarchies include a working file system sub-hierarchy (referred to throughout the present application as the working file system) and at least one alternate file system sub-hierarchy (referred to throughout the present application as the at least one alternate file system). Each sub-hierarchy includes a respective root area

through which other areas of the respective sub-hierarchy are accessible by direct or indirect connection and/or containment.

[0030] In a first embodiment, using one exemplary implementation, the working file system includes files and/or directories and their contents in the root directory, and files and/or directories and their contents accessible therefrom, e.g., branching therefrom, not including at least a directory herein referred to as the backup container directory, and files and/or directories accessible therefrom and their contents, e.g., branching therefrom. Essentially, the working file system includes the files and/or directories that would be typically included in a conventional HFS. The at least one alternate file system is accessible via the backup container directory, including files and/or directories and their contents accessible therefrom, e.g., branching therefrom. Corresponding root areas of the respective alternate file systems are accessible via the backup container directory. Each alternate file system typically corresponds to a different backup or a different configuration. In the present embodiment, the individual alternate file systems as well as the working file system do not overlap one another, i.e., are mutually exclusive. Preferably, the backup container directory does not overlap the working file system, but is linked to the working file system.

[0031] The at least one alternate file system is preferably in a hidden area to which access by a user and the working file system (including an operating system included therein) is denied without authorization. The backup container directory is preferably a hidden area to which access by a user is ordinarily denied. The backup container directory is typically limited to being accessed when privileged access is granted to one

or more alternate file systems, such as for an exchange process or updating of one or more alternate file systems, as described further below.

[0032] In another embodiment, using a different implementation, such as one that does not include the backup container directory, the sub-hierarchies may overlap one another, i.e., are not mutually exclusive. Furthermore, the at least two sub-hierarchies may be linked directly or indirectly to one another, including via overlapping portions or via non-overlapping portions. The respective root directories of the alternate file systems are located in locations other than the root directory of the HFS. The root directory of an alternate file system may be located in a location used (i.e., occupied) by another alternate file system or the working file system. When an exchange procedure is performed, the links are reconfigured. Any portion of the at least one alternate file system that is not included in the working file system is preferably hidden preventing access by a user and the working file system (including an operating system included therein) without authorization.

[0033] Other implementations are envisioned. For example, files and/or directories may be included that are accessible via the root directory but not included in the working file system or the at least one alternate file system. In another example, files and/or directories may be included that are accessible via the backup container directory, but are not included in the working file system or the at least one alternate file system.

[0034] With regards to either embodiment, the FSES module 16 is storable on a computer-readable medium, such as RAM, a hard drive, ROM, flash memory, a CD-ROM, a smart card, a diskette, etc., or transmitted by propagated signals. The series of

programmable instructions may be implemented in configurations including firmware, logic circuitry, software or a combination thereof.

[0035] The FSES module 16 is responsive to a request by a privileged user or the processor for reconfiguring the HFS to exchange a selected alternate file system of the at least one alternate file system with the current working file system, so that the selected alternate file system becomes the new working file system, and preferably, the current working file system becomes an alternate file system. Alternatively, the current working file system may be deleted or discarded.

[0036] The present invention is usable for many applications, including backups, multi-user environments, and system upgrading. For example, an alternate file system may include a backup copy of the working file system. Should a failure occur in a critical file of the working file system, the computer can be restored to full usefulness by exchanging the alternate file system including the backup copy with the working file system.

[0037] In another example, individual alternate file systems may be stored, each providing a respective customized environment, such as for use with multiple users. The individual alternate file systems may include different operating systems. Each user could then choose their respective customized environment by selecting an alternate file system to be the working file system.

[0038] In still another example, two of the sub-hierarchies may include identical contents. An upgrade is subsequently performed to one of the sub-hierarchies which is designated the working file system, while the other sub-hierarchy, which is designated an alternate file system, remains unchanged. Upgrades are notoriously prone to cause

problems. If the user accepts the upgrade, the alternate file system may be updated to include the upgrade. Alternatively, if the user rejects the upgrade, the working file system is exchanged with the unchanged alternate file system.

[0039] Exchange of the current working file system with the selected alternate file system in accordance with the present invention, herein referred to as a file system exchange, includes exchanging execution of operating systems and exchanging access to and association with the typically immutable root directory of the HFS and data included in and associated thereto. Advantages to the present invention include minimal down time when changing the file system, since actual copying or modification of data is minimized during the file system exchange. In addition, once the file system exchange operation is performed, it is reversible by the same process. Updating the contents of alternate file systems can be performed without interrupting the normal operation of the computer. With proper privileges, files within the alternate file systems may be accessed, which provides the ability to update selected files of a selected alternate file system, and to recover individual files without performing a file system exchange.

[0040] Since all of the sub-hierarchies use the same volume, system volume, pool of physical disk blocks, and/or a single data partition, the multiple copies of data occupy a minimal amount of space. Furthermore, file system maintenance applications may traverse the contents of the entire HFS, including all alternate file systems, thus maintaining integrity of all areas.

[0041] Referring to FIGS. 2A and 2B, with regard to the first embodiment, an exemplary HFS 200 is shown having a top most node configured as root directory 202, which is the true root directory of the HFS 200, i.e., all files and directories not included

in the root directory 202 are accessible, e.g., branch (directly or indirectly) from the root directory 202. HFS 200 is shown to include first, second and third exemplary sub-hierarchies 204a, 204b, 204c, where each sub-hierarchy includes a respective sub-hierarchy root node configured respectively as sub-hierarchy root directories 206a, 206b, 206c, and a plurality of files 208a, 208b, 208c including all files and files in all directories depending (i.e. branching) from the respective root directory 206a, 206b, 206c. In the exemplary implementation shown, specially designated directories and/or files, including a backup container directory 210 and other special files 212 (which may include files and/or subdirectories), accessible from the root directory 202, are not included in any of the sub-hierarchies 204a, 204b, 204c. Examples of special files 212 include a recycle bin, system volume information, volume labels and page file.

[0042] In FIG. 2A, HFS 200 is shown in a first state, state “A”, in which the sub-hierarchy root directory 206a is located in the root directory 202 located at location L1, and the other sub-hierarchy root directories 206b, 206c are located at other assigned locations L2, L3, respectively. The plurality of files 208a, 208b, 208c are configured to branch from their respective root directories 206a, 206b, 206c. Sub-hierarchy 204a is currently the working file system storing an operating system which is executing on a processor of the computer accessing the HFS, and sub-hierarchy 204a is being accessed by the operating system. Sub-hierarchies 204b and 204c are both alternate file systems descending from the backup container directory 210. Operating systems stored within the sub-hierarchies 204b and 204c are not currently executing. Special files 212 are accessible via the current working file system by the currently executing operating system.

[0043] FIG. 2B shows HFS 200 in a second state, state “B”, in which sub-hierarchy root directory 206c is located in the root directory 202, located at location L1 and the other sub-hierarchy root directories 206a, 206b are located at assigned locations L3, L2, respectively. The plurality of files 208a, 208c are re-configured to branch from their respective root directories 206a, 206c. Sub-hierarchies 204a and 204c have been exchanged, and sub-hierarchy 204c is currently the working file system. There is not a need for the special files 212 to be exchanged or copied. Instead, special files 212 are accessible via the current working file system after the exchange, i.e., by the currently executing operating system, which is the operating system stored by the current working file system, sub-hierarchy 204c. Sub-hierarchies 204b and 204a are both alternate file systems descending from the backup container directory 210. Operating systems stored within the sub-hierarchies 204b and 204a are not currently executing.

[0044] With respect to FIG. 3, a flowchart 300 is shown with exemplary steps executed by the file system exchange software module 16 for implementing the process of exchanging sub-hierarchies 204a and 204c, while the working file system is changed from 204a to 204c. The exchanging procedure may be initiated by a user request or by a processor request. Preferably access by the at least one processor 14 to the HFS 200 is prevented once execution of step 302 has been commenced through completion of execution of steps 302, 306 and 310, other than access needed to execute the steps 302, 306 and 310. At step 302, files 208a, i.e., all files and all files in all directories except the backup container directory 210 and special files 212 branching directly from the sub-hierarchy root directory 206a, which is currently located in the HFS root directory 202 while sub-hierarchy 204a is the working file system, are reconfigured to branch directly

from sub-hierarchy root directory 206c. Likewise, files 208c, i.e. all files and all files in directories branching from sub-hierarchy root directory 206c (not including those that were just reconfigured) are reconfigured to branch from the HFS root directory 202. A temporary storage location is provided to temporarily store pointers and other data being exchanged. At this point all of the files 208a, 208c of the sub-hierarchies 204a and 204c have been exchanged other than the respective root directories 206a and 206c.

[0045] At step 306, the sub-hierarchy root directories are exchanged, i.e., sub-hierarchy root directory 206a is moved to location L3, and sub-hierarchy root directory 206c is moved to location L1. The exchanging of the sub-hierarchy directories includes moving their contents and associated data such as parent pointers or special system files, depending on the actual HFS implementation used. The sub-hierarchy root directories are carefully altered in accordance with the file system implementation used to insure that the sub-hierarchy root directory 206c of the post-operation working file system includes features required for the root directory 202, and the sub-hierarchy root directory that was switched out of the root directory 202 no longer includes features exclusive to the root directory 202. Parent directories of the exchanged sub-hierarchy root directories are also updated to reflect the exchange. Exchanging working file systems using the FAT32 and NTFS HFS implementations are described in greater detail below with reference to FIGS. 4 and 5.

[0046] At step 310, file system data is updated to reflect the changes made to the root directory. For example the boot sector may be updated to record the fact that root location L1 now contains the sub-hierarchy root directory 206c.

[0047] The process of exchanging is preferably performed at an early stage in the startup (i.e., during booting) of the operating system. For additional protection from a hardware failure, power supply failure, etc., the exchange is preferably performed in a manner that insures that a readable file system exists substantially throughout the exchanging procedure. The exchanging procedure may alternatively be performed while an operating system associated with a current working file system is executing. Furthermore, with proper authorization selected files of the current working file system may be exchanged with selected files of one or more other sub-hierarchies rather than exchanging entire sub-hierarchies.

[0048] The exchange procedure includes exchanging positions of the sub-hierarchy root directories and reconfiguring the connections between the sub-hierarchy root directories to files/and or directories that are directly connected to them (not necessarily in the order specified) so that each sub-hierarchy root is connected to the appropriate data. Copying of data is minimized, since the data connected to each respective sub-hierarchy root is not copied. Furthermore, connections between files and/or directories holding the data connected to the respective sub-hierarchy roots do not need to be reconfigured.

[0049] With reference to FIG. 4, a flowchart is provided to illustrate steps performed during an exchange procedure of a current working file system which is a first sub-hierarchy and a second sub-hierarchy using the FAT32 file system implementation.

[0050] The root directory of the working file system (herein referred to as the “first root directory”) is located at the true root directory of the HFS, having starting cluster number “2” in this example. At the time that the second sub-hierarchy was created, the

starting cluster number of its root directory (herein referred to as the “second root directory”) was preferably stored for future use, such as during an exchange procedure. During an exchange procedure, the starting number of the second root directory may be retrieved from storage or may be interpreted by the HFS. If the HFS has been damaged or corrupted and is unable to be interpreted, the stored number may be retrieved. In this example, the starting cluster number of the second root directory is “X”.

[0051] At step 404, a determination is made for each file stored in the root directory of the working file system to determine if the file is a subdirectory. If so, the parent pointer for the subdirectory is changed from “2” to “X”. Ordinary files do not need modifications as they do not have a parent pointer.

[0052] At step 408, a similar procedure is followed for the files stored in the second root directory. The parent pointer for each subdirectory branching from the root directory of the alternate file system is changed from “X” to “2”. At this point, all of the files branching from the first root directory have been exchanged with all of the files branching from the second root directory.

[0053] At step 420, the volume label (if it exists) of the first root directory is moved to the second root directory.

[0054] At step 424, self and parent pointers from the second root directory are removed, and valid self and parent pointers are added to the first root directory. The self pointer value is “2”, and the parent pointer value is copied from the original value of the second root directory’s parent pointer value (i.e. pointing to the backup container directory).

[0055] At step 428, the second root directory's parent directory (the backup container directory) is updated so that the original entry for the second root directory is changed from "X" to "2", so that the backup container directory now points to the first root directory.

[0056] At step 432, the boot sector is updated so that the first root directory's cluster number "2" is replaced with the second root directory's cluster number "X". This causes the computer to reference the second root directory while booting. The second root directory is now the true root directory of the FAT32 HFS.

[0057] With respect to FIG. 5, a flowchart showing procedural steps of an exemplary NTFS (New Technology File System) implementation is provided. A first sub-hierarchy is provided having a first root directory occupying file record number "5", where first sub-hierarchy is the current working file system. A second sub-hierarchy, the alternate file system, is provided having a second root directory occupying file record number "Y", where "Y" is stored upon formation of the second sub-hierarchy and retrieved for an exchange procedure, and/or the file record number "Y" is obtained by interpreting the second sub-hierarchy. The first root directory is the true root directory of the HFS, and includes, in predetermined file records, a plurality of special system files for storing housekeeping data. The second root directory includes a placeholder file to correspond to each respective special system file stored in the first root directory, where each placeholder file has the same filename as the corresponding special system file. A placeholder file is also provided for the root directory name itself, which is named ".", ("dot"), which is an illegal filename otherwise, since it is reserved for the self-pointer of all directories. A placeholder of a similar name may be used, such as "..." ("dot-dot-

dot”). Lastly, a placeholder file is provided for the backup container directory. The placeholder files are not used, and exist only to provide directory entries which maintain proper collation (sorting order) within the root directories.

[0058] The following description of the steps included in the flowchart of Fig. 5 uses NTFS nomenclature, as known to those skilled in the art. At step 504, each file entry in the first and second root directories other than the system files and the placeholder files are modified by exchanging their owning directories, i.e., the parent fileref field of each entry is changed from the first root directory to the second root directory, and vice versa. At step 508, the file records are modified for each file entry in the first and second root directories by exchanging the owning directory’s back-pointers for the other directory’s back-pointers, e.g., the file’s FILE record’s matching file info entry’s parent fileref is exchanged for the parent fileref of the other root directory. It is possible that the affected file info entry is not located in the FILE record itself, but in a chained subrecord. In this case, both the subrecord and the main record are altered. Preferably, modifications made in steps 504 and 508 are performed using fixed-size fields so that the lengths of the directory records and/or the FILE records will not change.

[0059] At step 512, the FILE record of each respective special system file of the plurality of special system files is exchanged with the FILE record of the corresponding placeholder file. Preferably, the backup container directory is treated as a special system file, and exchanged accordingly. At step 516 the directory attribute data (directory listing) in the FILE records are exchanged between the first and second root directories, e.g., the \$I30(directory) \$INDEX_ROOT, \$INDEX_ALLOCATION and \$BITMAP

attributes are exchanged between the FILE records of the first and second root directories, performing the actual exchanging thereof.

[0060] It is possible that the combined lengths of the previously mentioned attributes differ between the two FILE records being exchanged. To accommodate this situation, the FILE record having a shorter length is grown (lengthened) before the exchange, and the other FILE record is shrunk (reduced in length) after the exchange. Furthermore, it is possible that the internal attribute-IDs associated with the previously mentioned attributes will differ between the two FILE records being exchanged. If so, either each attribute's attribute-ID is maintained during the exchange, or all attribute IDs (and the next-attribute-ID field) are recalculated for both FILE records being switched.

[0061] Steps 504-512 are preferably performed by walking through both directories in lockstep and treating each entry in accordance with steps 504-512 before proceeding to the next entry. An entry with a subnode flag is eight bytes longer than a matching entry without the subnode flag. Preferably, these last eight bytes are not exchanged, the subnode flag is not exchanged and is preserved, and the entry length word is not exchanged and is preserved. The FILE records and directory index records preferably use the conventional NTFS write-completion detection fixup scheme.

[0062] The NTFS implementation described, unlike the previous FAT32 implementation, does not require any modifications to the boot sector.

[0063] It will be understood that various modifications may be made to the illustrative embodiments disclosed herein. Therefore, the above description should not be construed as limiting, but merely as exemplifications of preferred embodiments. Those skilled in

the art will envision other modifications within the scope and spirit of the claims appended hereto.